Carnegie Mellon
Software Engineering Institute

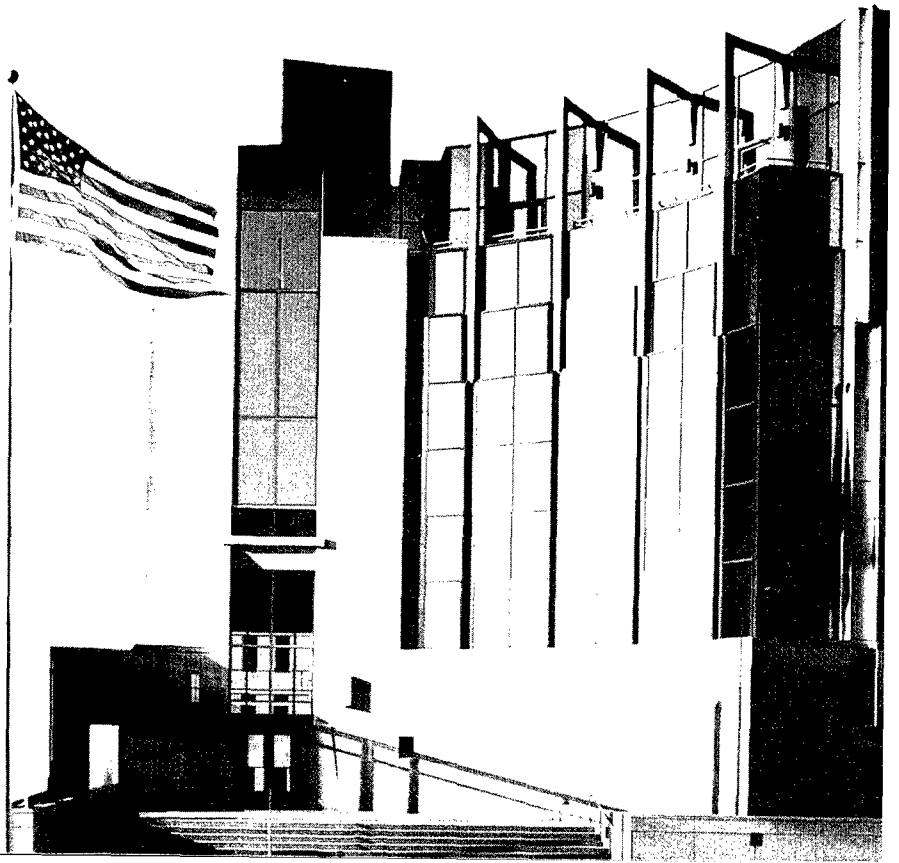
Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview

Key Elements in Building, Fielding, and Supporting Commercial-off-the-Shelf (COTS) Based Solutions

Authors:
Cecilia Albert, Software Engineering Institute
Lisa Brownsword, Software Engineering Institute

In Collaboration with:
Colonel David Bentley, USAF
Thomas Bono, MITRE
Edwin Morris, Software Engineering Institute
Debora Pruitt, MITRE

July 2002



TECHNICAL REPORT
CMU/SEI-2002-TR-009
ESC-TR-2002-009

Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate in admission, employment, or administration of its programs or activities on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973 or other federal, state, or local laws or executive orders.

In addition, Carnegie Mellon University does not discriminate in admission, employment or administration of its programs on the basis of religion, creed, ancestry, belief, age, veteran status, sexual orientation or in violation of federal, state, or local laws or executive orders. However, in the judgment of the Carnegie Mellon Human Relations Commission, the Department of Defense policy of "Don't ask, don't tell, don't pursue" excludes openly gay, lesbian and bisexual students from receiving ROTC scholarships or serving in the military. Nevertheless, all ROTC classes at Carnegie Mellon University are available to all students.

Inquiries concerning application of these statements should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-6684 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-2056.

Obtain general information about Carnegie Mellon University by calling (412) 268-2000.

Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview

Key Elements in Building, Fielding, and Supporting Commercial-off-the-Shelf (COTS) Based Solutions

CMU/SEI-2002-TR-009
ESC-TR-2002-009

Authors:
Cecilia Albert, Software Engineering Institute
Lisa Brownsword, Software Engineering Institute

In Collaboration with:
Colonel David Bentley, USAF
Thomas Bono, MITRE
Edwin Morris, Software Engineering Institute
Deborah Pruitt, MITRE

July 2002

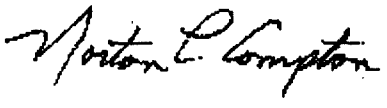
COTS-Based Systems Initiative

This report was prepared for the

SEI Joint Program Office
HQ ESC/DIB
5 Eglin Street
Hanscom AFB, MA 01731-2116

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

FOR THE COMMANDER



Norton L. Compton, Lt Col, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense. The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense.

This work was created in the performance of Federal Government Contract Number F19628-00-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 52.227-7013.

For information about purchasing paper copies of SEI reports, please visit the publications portion of our Web site (http://www.sei.cmu.edu/publications/pubweb.html).

Table of Contents

List of Figures

Acknowledgements

Colonel David P. Bentley, Director, Business Information System Program Office, Electronic Systems Command (ESC/MM), United States Air Force provided the vision for this project. He also provided some of the funding, and much of the impetus that kept the project moving. Thomas L. Bono and Debora M. Pruitt of MITRE Corporation provided a sounding board for the major concepts, and invaluable editorial support. Edwin Morris of the Software Engineering Institute (SEI) clarified our complex thoughts and developed much of the material on COTS component evaluation.

John Foreman and Thomas Brandt supported this project as it became more complex. David Carney, Patricia Oberndorf, and Patrick Place provided valuable insights into the unique aspects of a process framework for COTS-based systems and comments that improved the quality of this document. Fred Long, Marc Kellner, and Fred Hansen of the SEI, and Judy Clapp, Mike Bloom, Pam Martin, and Dave Hart of MITRE Corporation read versions of this document. Their comments made this document much stronger.

Abstract

Government and private organizations are escalating their use of commercial off-the-shelf (COTS) and other pre-existing components in critical business systems. Attempts to exploit these components through use of traditional engineering approaches that involve defining requirements, formulating an architecture, and then searching for components that meet the specified requirements within the defined architecture have been disappointing.

The Evolutionary Process for Integrating COTS-based systems (EPIC)* redefines acquisition, management, and engineering practices to more effectively leverage the COTS marketplace and other sources of pre-existing components. This is accomplished through concurrent discovery and negotiation of diverse spheres of influence: user needs and business processes, applicable technology and components, the target architecture, and programmatic constraints. EPIC codifies these practices in a structured flow of key activities and artifacts. This alternative approach is a risk-based, disciplined, spiral-engineering approach which leverages the Rational Unified Process® (RUP®).

This document is the first release of an overview of the EPIC framework along with its activities and artifacts. The first release of the full description of EPIC is found in the Software Engineering Institute technical report: Evolutionary Process for Integrating COTS-Based Systems (EPIC).[1] These documents will be updated based on reader's comments and lessons learned from use of EPIC.

* Also known as Information Technology Solutions Evolution Process and Integrating Technology by a Structured Evolutionary Process (ITSEP).

® Rational, RUP, and Rational Unified Process, among others, are trademarks or registered trademarks of Rational Software Corporation in the United States and/or in other countries.

[1] To Be published. *Evolutionary Process for Integrating COTS-Based Systems (EPIC)* by Cecelia Albert and Lisa Brownsword.

Chapter

Overview

Use of commercial off-the-shelf (COTS) and other pre-existing components[2] is gaining popularity, particularly where the organization's needs match those of one or more commercial information technology (IT) marketplace segments. There is a vibrant market today that delivers COTS software components that range from software development environments to operating systems, database management systems, and increasingly, business and mission applications. COTS components and, to a lesser extent, other pre-existing components, offer the promise of rapid delivery to the end users, shared development costs with other customers, and an opportunity for expanding mission capabilities and performance as improvements are made in the marketplace. Few organizations today can afford the resources and time to replicate market-tested capabilities.

Yet, the promise of using pre-existing components is too often not realized in practice [1, 2, 3]. Many organizations find that COTS-based systems are difficult and costly to build, field, and support. A major cause of this difficulty is that organizations building these systems tend either to assume that components can be simply thrown together or they fall back on the traditional engineering skills and processes with which they are familiar–skills and processes that have been shown not to work in the building of a COTS-based system [4].

Experience [5] shows that the effective use of COTS components demands a new way of doing business: new skills, knowledge, and abilities; changed roles and responsibilities; and different processes–and these changes are not happening. To achieve the benefits of the commercial marketplace while managing the drawbacks, the project must drive an evolving definition of the requirements, the end-user business processes, the architecture, and the cost, schedule, and risk as more is learned about the capabilities of the available COTS components.

[2] Pre-existing components include hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (a piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware).

Scope

The Evolutionary Process for Integrating COTS-based Systems (EPIC)[3] was developed to help organizations build, field, and support solutions based on COTS and other pre-existing components. Components, as the term is used in EPIC, includes hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (a piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware).

In EPIC, the fundamental concept is to build, field, and support a *solution* that provides important, useful capability to the organization. Ideally, the scope of a solution is defined so that it can be initially fielded in a period of 6 to 12 months. Many organizations' needs exceed this scope. Where possible, these needs are decomposed into many solutions to be developed in sequence or in parallel–or both. In this case, each solution replaces or augments already fielded solutions with enhanced and added functionality.

A solution integrates

- one or more pre-existing hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (a piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware)
- any required custom code (including wrappers and "glue")
- appropriate linkage/interface to the broader organization's architecture and external systems
- end user's business processes including any changes necessary to match the processes provided by the integrated components and custom code

EPIC integrates COTS lessons learned, disciplined engineering practice, and end-user business process change management to build, field, and support a solution. EPIC may be used for any solution that is controlled and directed through central project management using one or more teams of engineers.

Target Audience

This document is written for the stakeholders responsible for building, fielding, and supporting solutions based on COTS and other pre-existing components who want a basic understanding of EPIC. Stakeholders who are ready to apply EPIC should refer to the companion document that contains a full description[4].

For EPIC, stakeholders include the broadest possible group of experts who understand and are empowered to help define

[3] Also known as Information Technology Solutions Evolution Process and Integrating Technology by a Structured Evolutionary Process (ITSEP).

[4] Evolutionary Process for Integrating COTS-Based Solutions (EPIC): Building, Fielding, and Supporting Commercial-off-the-Shelf (COTS) Based Solutions. CMU/SEI-2002-TR-005.

- the mission need [e.g., end users (operators, database administrators, installers), business process owners, customers, sponsors, and assorted subject matter experts]
- systems engineering, integration, and maintenance (e.g., engineers, architects, developers, testers)
- contract and cost analysis (e.g., project office personnel)
- the marketplace and other component sources (e.g., vendors/suppliers, technology experts, industry consultants, legacy maintainers)
- changes to end-user business processes

To be effective, the stakeholders must form a team where each member is committed to sharing individual expertise and is willing to compromise some expectations to accommodate the capabilities of the available components. EPIC provides a framework that links the disparate stakeholders into a coherent team that simultaneously defines and manages tradeoffs among requirements and end-user business processes, system architecture and design, and programmatic factors such as procurement strategy, end-user business process change management, cost, schedule, and risk.

Origins

EPIC evolved from a U.S. Air Force need to meet the challenges of building, fielding, and supporting COTS-based business solutions. To meet this need, EPIC builds on and integrates the work of many others:

- The Software Engineering Institute (SEI) COTS-Based Systems Initiative has done extensive research on processes needed in the management of COTS-based systems [3,4,6,7,8], engineering techniques for designing and evolving COTS-based systems [9], and evaluation techniques for assessing: COTS-based program risks, suitability of COTS products, and appropriateness of COTS-based system designs[5].
- The Rational Unified Process (RUP) [10,12] provides a disciplined, risk-based spiral engineering approach that extends the work of Dr. Barry Boehm [11]. The EPIC phases, anchor points, and most artifacts, terms and descriptions are from the RUP. The RUP also provides essential project management and engineering support activities such as project planning, quality assurance, requirements management, and configuration management.

[5] In addition to the publicly available work, this document builds on *COTS-Based Systems for Program Managers* (tutorial) by L. Brownsword, P. Oberndorf, and C. Sledge; *COTS Product Evaluation* (tutorial) by P. Oberndorf, J. Dean, E. Morris, and S. Comilla-Dorda; *COTS Usage Risk Evaluation* (CURE) by D. Carney, P. Place, and E. Morris; and *Acquisition/Assembly Process for COTS-Based Systems* by D. Carney, P. Oberndorf, P. Place, L. Brownsword, and C. Albert.

Reader's Guide to This Document

This document is designed to explain the basic principles of EPIC and is organized in the following chapters.

- Chapter 2 provides insight into the challenges of using pre-existing components, particularly COTS components. These issues form the requirements for lifecycle processes that underlie the definition of EPIC.
- Chapter 3 summarizes the EPIC framework.
- Chapters 4 through 7 contain summary descriptions of each of the four EPIC phases: Inception, Elaboration, Construction, and Transition. For each phase, the goal, objectives, exit criteria, and activities are listed.
- A Glossary of terms related to EPIC, and a list of the EPIC References is provided at the end of the document.

This overview and the full description of EPIC[6] the first release of EPIC documentation. These documents will be updated based on reader's comments and lessons learned from use of EPIC. Comments or suggestions about the documents are welcome and examples of use of EPIC are solicited. Comments, suggestions, and examples should be sent to the authors at

cbs@sei.cmu.edu

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

[6] To be published. *Evolutionary Process for Integrating COTS-Based Systems (EPIC)* by Cecilia Albert and Lisa Brownsword.

Process Drivers

Using pre-existing components, particularly COTS components, introduces significant challenges. These challenges drive the life cycle processes comprise EPIC.

Building solutions based on incorporating pre-existing components is different from typical custom[7] development in that the components are not designed to meet a project-defined specification. COTS components are built to satisfy the needs of a market segment. Therefore, an understanding of the components' functionality and how it is likely to change over time must be used to modify the requirements and end-user business processes as appropriate, and to drive the resulting architecture.

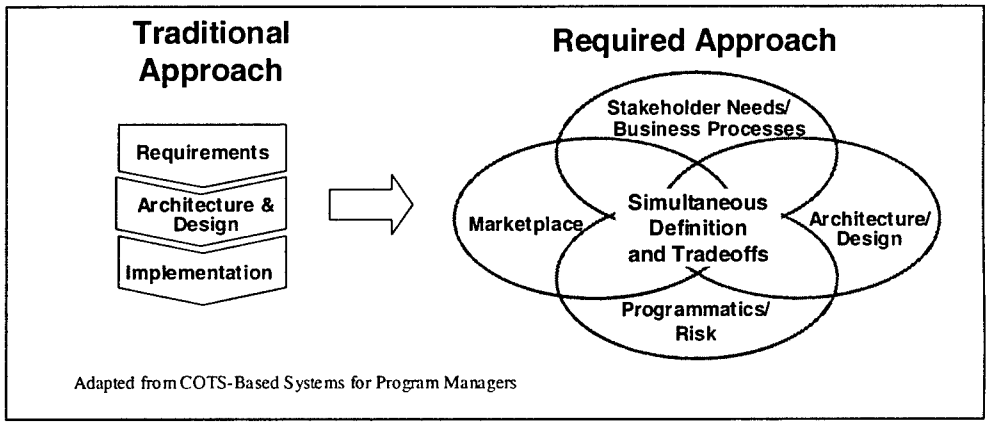


Figure 1. Required Approach for COTS-Based Systems

Key to building solutions based on components is the need to simultaneously define and make tradeoffs among four *spheres of influence*[8], as shown in Figure 1. While tradeoffs are common in any engineering endeavor, tradeoffs in this case are driven by the desire to leverage components from the marketplace. This is a fundamental change[9] from alternative process models. Numerous projects have unsuccessfully tried to integrate pre-existing

[7] By *custom* development we mean uniquely built in direct response to a buyer's specifications.

[8] *Sphere of Influence* is used to represent a set of information with common or related stakeholders, techniques for gathering and managing the information, and the dynamic by which the information changes.

[9] *COTS-Based Systems for Program Managers* (tutorial); L. Brownsword, P. Oberndorf, and C. Sledge.

components using the more traditional approach (shown on the left) consisting of defining the requirements, then formulating an architecture to meet those requirements, and then trying to fit components into that architecture.

Instead as shown on the right of Figure 1, an emphasis on balance between four spheres of influence is critical throughout the life of a project. The four spheres represent competing interests that must be considered in forming a viable solution that effectively leverages pre-existing components:

- Stakeholder needs and business processes denotes requirements (including quality attributes such as performance, security, and reliability), end-user business processes, business drivers, and operational environment.
- Marketplace denotes available and emerging COTS technology and products, non-development items (NDI), and relevant standards.
- Architecture and Design denotes the essential elements of the system and the relationships between them. The elements include structure, behavior, usage, functionality, performance, resilience, reuse, comprehensibility, economic and technologic constraints and tradeoffs, and aesthetic issues. [12]
- Programmatics and risk denotes the management aspects of the project. These aspects consider the cost, schedule, and risk of building, fielding, and supporting the solution to include the cost, schedule, and risk for changing the necessary business processes.

These four spheres are simultaneously defined and traded through the life of the project because a decision in one sphere will inform and likely constrain the decisions that can be made in another sphere. For example, a stakeholder need may be stated in a way that cannot be satisfied by any pre-existing component. Similarly, a potential pre-existing component may not be compatible with the organization's existing infrastructure or use a licensing strategy that would be cost prohibitive. Further, the new release of an already selected component may change the behavior of the solution.

Commercial Marketplace-Imposed Constraints

The unique characteristics of COTS components introduce dynamics and specific constraints that must be accommodated. COTS components[10] are

- sold, leased, or licensed to the general public
- offered by a vendor trying to profit from them
- supported and evolved by the vendor, who retains the intellectual property rights
- available in multiple, identical copies
- used without modification of the internals

[10] *COTS-Based Systems for Program Managers* (tutorial); L. Brownsword, P. Oberndorf, and C. Sledge.

Based on these characteristics, the SEI has identified the following attributes [6] of COTS components:

- The marketplace[11], not one system's needs, drives COTS component development and evolution.
- COTS components and the marketplace undergo frequent, almost continuous change.
- Frequency and context of COTS component releases are determined at the discretion of the vendor.
- COTS components are built based on unique architectural assumptions and are not constructed using a universal or consistent architectural paradigm.
- There is at best limited visibility into COTS component internals and behavior.
- COTS component assumptions about end-user processes may not match those of a specific organization.
- "Vendor" is not a new name for subcontractor. Different relationships are required to have insight and to influence component changes.
- COTS components often have unsuspected dependencies on other COTS components.

Derived Requirements

Any process that builds, fields, and supports solutions must

- create an environment where components and the marketplace drive the evolving definition of the solution. EPIC reflects the reality that the ultimate control of critical components has passed from the hands of the project to those of the component suppliers. EPIC focuses on reconciling the diverse expectations of stakeholders with an evolving understanding of the capabilities of available components. Therefore, an environment that facilitates hands-on analysis of the components and continuous negotiation with stakeholders will be required for the life of the project to evaluate new and changed components and their impacts on evolving solutions.
- compose solutions from a diverse range of components. Solutions are built from a combination of components, both hardware and software–and many components are themselves composed from components. Insight into the inner workings of these components will vary (e.g., black, white, and gray box) depending on the source and the intended use of the component. Engineers must infer the behaviors of various component combinations as they integrate the components they buy (and otherwise procure). Hands-on experience is essential with any component critical to the success of the solution in an environment that represents, as closely as possible, the operational use of the component.

[11] *Marketplace* refers to the aggregation of buyers and sellers where goods are offered for sale, lease, or license.

- implement disciplined spiral or iterative systems engineering practices. Spiral development allows the discovery of the critical attributes of the solution through an evolutionary exploration of the highest risk elements of the system and the components available to address them. It also allows frequent and direct feedback from all of the affected stakeholders through evolving prototypes that characterize and mature the architecture while reducing the risk in the solution.

- support concurrent and integrated implementation of engineering, business, and procurement activities. The volatility of the COTS marketplace means that decisions about the COTS components used, the structure that accommodates them, and the associated cost, schedule, and risk of the project are made continuously through the life of the solution. In addition, an engineering decision to include a new component is also a decision to acquire the component from its supplier. The engineering processes, management processes, oversight processes, and procurement processes of the project must be coordinated to support the flexibility and negotiation of requirements and iterative definition of the solution.

- balance component obsolescence and solution stability. Due to the volatility of the marketplace, a new component or a new release of a component being used can be introduced at any time during building, fielding, or support of the solution. Engineers must continuously monitor the marketplace through the life of the project to anticipate the changes being made. An environment where new components and releases can be evaluated and their potential impacts assessed is required.

- accommodate a broad spectrum of ways components comprise solutions. Different components can be used in a variety of ways to form a spectrum of possible solutions. Some solutions are composed of a single component or a vertically integrated set of components from one vendor (components designed to work together–e.g., Microsoft Office). The focus in this case is on tailoring the component to work in the target organization. At the other end of the spectrum are solutions composed of multiple components from multiple vendors that must be integrated to provide system functionality. This kind of solution may contain significant components that are custom developed and/or pieces of legacy systems and may require significant effort to integrate.

- develop and maintain a flexible system architecture as a centrally managed asset. Since the components are "owned" by the suppliers, the framework by which the components are linked to support the organization's needs–the architecture–becomes a key organization asset. With COTS-based systems, continuous, rapid changes driven by new mission needs, component upgrades, and technology advances are facts of life. An architecture that can retain its structure and cohesiveness yet allow the system to respond easily to these changes–an evolvable architecture–becomes an important strategic asset to an organization.

- design *with* components versus design *for* reuse. Building a solution is inherently an act of composition and reconciliation. Designers start with a general set of needs and explore the offerings of the marketplace to see how closely they match the needs, then

design an architecture around the pre-existing components. Building systems for reuse, on the other hand, has come to mean designing and building structures and components that can be used again in related systems. In this case, the designers have direct visibility and control of the components. The nature, timing, and order of the activities and the processes used differ accordingly.

- incorporate activities for changing the end user's business processes. Using components in solutions is not compatible with simply automating a predefined set of business processes. Since components embody the vendor/supplier view of end-user business processes, changes to the end user's business processes must be negotiated based on those inherent in the components under consideration. In EPIC, engineering activities are coordinated with activities for changing the end user's business process to ready the end-user community for fielding of the solution. The risks and implications of changes to the organizations where the solution is fielded may be significant drivers in the project. EPIC users must coordinate the definition and implementation of end-user business process and organizational changes through the life of the project.
- maintain and document how the component supports the solution. Many components have diverse functionality, not all of which is required in a given solution. It is important to document the functionality applicable to the solution. It is just as important to capture how any excess functionality is handled within the solution–especially any functionality that is blocked or bypassed. Guidance and training on any solution-specific use of the application will have to be considered as part of the deployment artifacts. Furthermore, after fielding, some of these "excess" capabilities will find their way into operational use. The ways in which the components are actually used within the organization must be tracked and captured. This information is important to the evaluation of new component releases after solution fielding, as vendors may make changes in segments of the component that were not originally considered part of the solution.

Framework

To accommodate the continuous change induced by the COTS marketplace, EPIC uses a risk-based spiral development process. EPIC users manage the gathering of information from the marketplace and the stakeholders and refine that information through analysis and negotiation into a coherent, emerging solution that is embodied in a series of executable representations through the life of the project. Stakeholders actively participate in EPIC as key players in day-to-day negotiations that also continue through the life of the solution. This also ensures their buy-in to the emerging solution.

EPIC is more than a way to select a specific component. Use of EPIC begins with the definition of a need for a new or changed capability and a commitment to provide the resources necessary to identify, acquire, build, field, and support a solution that will deliver that capability. Use of EPIC ends only when the solution is retired or replaced with a new solution. In some instances, the solution will be transitioned to a different organization for support once it has been fielded. The major features of EPIC should also be used by the support organization to protect the investment in components.

Iteratively Converging Decisions

In order to maintain balance between the four spheres, EPIC creates an environment that supports the iterative definition of the four spheres over time while systematically reducing the trade space within each. This allows a decision in one sphere to influence, and be influenced by, decisions in the other spheres.

Initially, as shown at the left of Figure 2, the trade space may be large. There is flexibility for making tradeoffs between the stakeholder needs and end-user business processes, the architecture and design, the offerings of the marketplace and other sources, and programmatics and risk. As EPIC is used to drive toward a refined understanding of the solution, the trade space shrinks. The spheres increasingly overlap as fewer decisions remain in any single sphere that can be made without significant impact on the others.

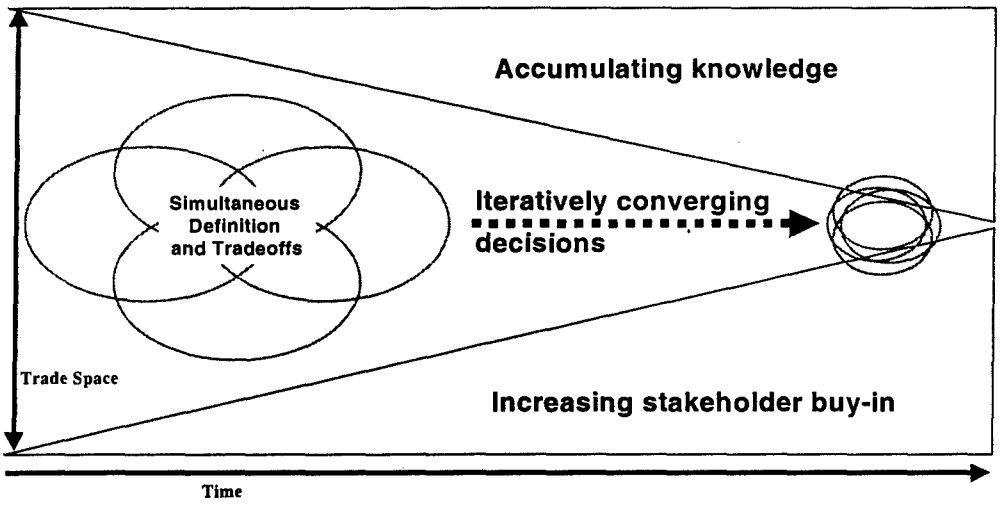


Figure 2. The EPIC Objectives

Accumulating Knowledge

Concurrent with diminishing the trade space, knowledge about the solution must grow at a controlled pace. This knowledge is reflected in the set of artifacts[12] or work products necessary to build, field, and support the solution. Most of the artifacts are started in outline form and are expanded as more information is gathered and refined.

This growing knowledge includes an increasingly detailed understanding of the following:

- capabilities and limitations of candidate components, the suppliers that produce them, and the marketplace drivers that control them
- negotiated and prioritized stakeholder needs and end-user business processes
- architectural alternatives and integration mechanisms that bind the components together
- implications of the components on the stakeholder needs, the end-user business processes, and the solution architecture
- planning necessary to implement and field the solution (including any needed end-user business changes) and associated cost, schedule, and risk

Keeping knowledge current about the components critical to the solution and the marketplace or other sources that supply components is particularly important. This allows the organization to track trends that may affect the solution over time and keep the volatility of the marketplace in balance with the need for stability for building, fielding, and supporting the solution in operations. Monitoring and evaluation begins at project initiation and continues until the solution is retired.

[12] Artifacts and work products may take a number of forms. Generally, they are not documents.

In some (unfortunate!) cases, marketplace events may invalidate earlier decisions (e.g., support for a component is dropped, a new component is introduced, or a feature is added to the component). While there is no easy resolution for such disruptions, early warning of impending changes will allow alternative decisions to be made in a deliberate and careful way; perhaps even before the trade space narrows.

Increasing Stakeholder Buy-in

While decisions are converging and knowledge is accumulating, the stakeholders must increase commitment to the evolving definition of the solution–and these stakeholders are a broad and possibly disparate group. This will be very difficult for many projects as this commitment is significant, even unprecedented. Active participation from the stakeholders, however, is essential to the success of the solution. Creating an environment that includes the stakeholders (or empowered representatives) directly affected by any change in end-user business processes allows EPIC to quickly resolve discovered mismatches between the available components, the desired end-user business processes, and the stated stakeholder needs while simultaneously demonstrating that the solution can be built within cost and schedule constraints with acceptable risk.

With increased understanding of available components, end-user needs mature and change. The day-to-day involvement of end users is essential to EPIC because the activities that identify, evaluate, and select components will shape the end-user business processes and define the functionality that will be delivered. At the same time, engineering stakeholders ensure that the components considered can be effectively integrated within the broader organization's existing systems to meet required performance parameters. Business analysts must ensure that viable suppliers support the components. Supplier involvement can provide enhanced visibility into the component's capabilities and potential insight for the suppliers into the organization's needs. The continuous negotiation and reconciliation among affected stakeholders leads to a more effective use of components in satisfying the mission.

The stakeholders confirm and increase their buy-in and commitment to the evolving definition of the solution based on an iteratively evolving executable representation of the solution. An executable representation is essential to reduce the risks due to misunderstandings or unforeseen technical and operational factors.

Evolution Through Iterations

EPIC uses a risk-based spiral development process to keep the requirements and architecture fluid as the four spheres of influence are considered and adjusted to optimize the use of available components. Iterations systematically reduce the trade space, grow the knowledge of the solution, and increase stakeholder buy-in. At the same time, each iteration, or spiral, is planned to mitigate specific risks in the project.

Each iteration in EPIC, as shown in Figure 3, begins with the development of a detailed *plan* for the iteration and ends with *assessing* whether or not the objectives in that plan were met. Iteration planning uses the current understanding of risk to establish goals and objectives, and defines the specific tasks as well as the cost, schedule, and resources needed for the iteration.

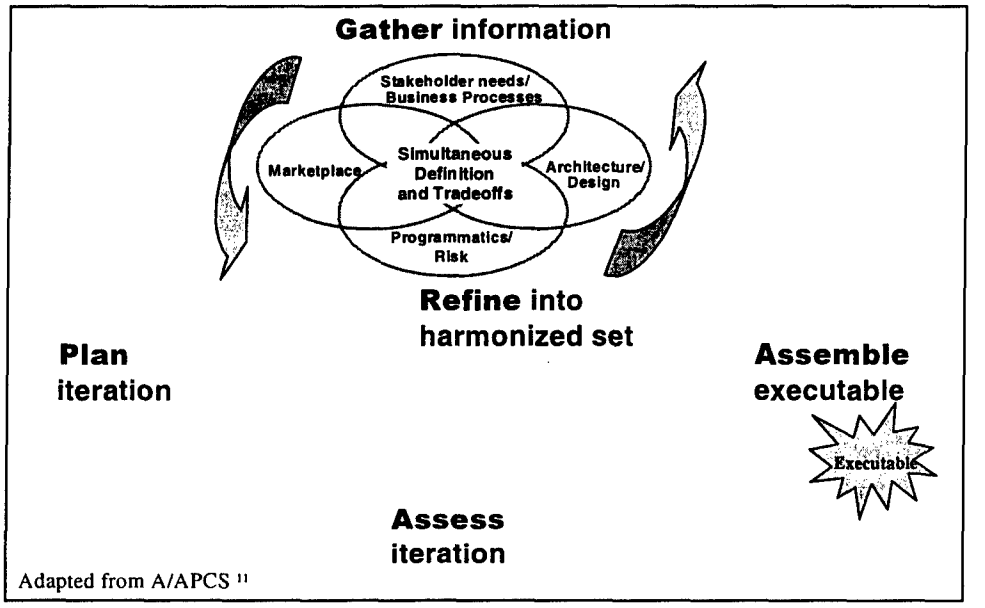


Figure 3. An Iteration in EPIC

Unique to building solutions are a number of inherently chaotic activities[13]. These are the activities that continuously *gather* information from each of the four spheres and *refine* the newly gathered information, through analysis and negotiation with affected stakeholders, to form the harmonized knowledge needed to *assemble* an Executable Representation of the solution. This knowledge is captured in artifacts that start at a very high level in early iterations and are expanded through cycles of gathering and refining across iterations as increasingly detailed information is harmonized. It may take many cycles of gathering and refining information within an iteration to produce knowledge sufficiently detailed and harmonized across the four spheres to meet the iteration objectives.

Every iteration assembles an Executable Representation of the solution that exhibits the common understanding of the solution that has been achieved among affected stakeholders to that point and demonstrates the adequacy of the solution to meet the iteration objectives.

While these activities are the same for every iteration, the focus, depth, and breadth of the activities within an iteration are adjusted to meet specific iteration objectives. Each of these activities is further described below.

[13] *Acquisition/Assembly Process for COTS-Based Systems (A/APCS)*; Carney D.; Oberndorf P; Place P.; Brownsword, L.; and Albert, C. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. To be published.

Plan

Planning is a continuous activity and the planning artifacts are updated with every iteration to reflect an evolving understanding of the solution. Planning occurs at two levels. The first level of planning creates (in the first iteration) and maintains (in each subsequent iteration) a coarse-grained plan for the project. This plan, the Development Plan, describes the capability that will be built and fielded and lays out, in general terms, the number of iterations, key milestones, and resources needed to mitigate risk and deliver the current solution.

The second level of planning creates a fine-grained plan for the current iteration. The Iteration Plan defines the goals and objectives for the iteration and determines the resources (including cost and schedule) necessary to meet them. The objectives for any given iteration are designed to meet high-priority functional needs and to mitigate high-priority risks to the project. (Note: The greatest risk posed to the project will likely be implementing the changes to the end-user business processes to match those in the available components.) Each iteration should be constrained to the amount of work that can be done in a fixed, relatively short (measured in weeks) period of time. Later iterations will build on the results of earlier iterations to produce the desired solution.

Gather and Refine

The gather activities collect the information needed to meet the iteration objectives from each of the four spheres of influence and build representations of the information specific to the type of information gathered. The refine activities synthesize the gathered information through analysis targeted at identifying incomplete information and identifying mismatches among the competing constraints and the potentially divergent classes of expectations. Incomplete information is resolved through gathering additional information. Mismatches are resolved through negotiation between the affected stakeholders. Where possible, end-user business processes are modified and requirements negotiated to allow greater use and leverage of available components.

While it is useful to think about information from these spheres individually because of the different techniques associated with gathering information from each of the spheres, the information gathered from one sphere depends on and drives the information needed from another sphere. In practice, this drives the practitioner to gather a little, refine a little, then gather some more, and then refine some more. For example, examination of a component may suggest that end users must select from a number of alternative security strategies. This may require an additional round of stakeholder needs elicitation to determine stakeholder security preferences.

The gather and refine activities produce harmonized artifacts that represent the current agreed-upon state of the solution and include all of the known data and previously accepted compromises necessary to meet the iteration objectives. Mismatches between information from the four spheres are remediated in successive gather and refine cycles using progressively more detailed information until it is determined that all information is sufficiently detailed to meet the objectives for the iteration at hand. It is important not to

allow information in one sphere to be gathered at too low a level of detail relative to the information in the other spheres. This tends to close out opportunities for trades across the spheres.

Assemble

An essential activity in every iteration is the effort to assemble one or more Executable Representations of the current agreed-upon state of the solution. In early iterations, the Executable Representation may be a mock-up of critical stakeholder needs. In later iterations, the Executable Representation is an evolutionary prototype that reflects the architecture and evolves to become the fielded solution. This prototype includes an ability to test the necessary infrastructure and any other systems with which the solution must interact. In addition, the Executable Representation for each phase must be sufficient to prototype the end-user business processes inherent in the solution.

Assess

The assess activities review the iteration to determine whether or not the iteration's objectives were achieved. In addition, a summary of what was learned and what more needs to be learned to mitigate risk is created. These activities also determine whether or not the stakeholders are still committed to proceed to the next iteration.

Phases Focus Iterations

The four spheres continuously evolve through successive iterations. Each iteration is designed to meet specific objectives and will nominally take one to eight weeks to complete. EPIC iterations are managed, as shown in Figure 4, by the four RUP phases (Inception, Elaboration, Construction, and Transition) and associated anchor points[14] (Life-cycle Objectives, Life-cycle Architecture, and Initial Operational Capability).

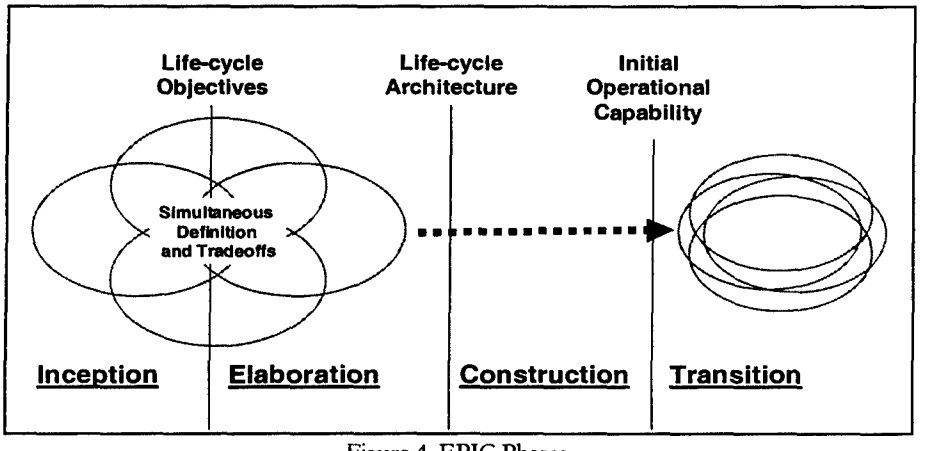


Figure 4. EPIC Phases

[14] RUP calls these lifecycle milestones.

Each EPIC phase consists of one or more EPIC iterations. Iterations in each phase build on and strengthen stakeholder understanding of the available components and each component's impact on requirements and end-user business processes, architecture and design, and the cost, schedule, and risk of implementing the solution.

The EPIC phases accommodate the continuous change induced by the COTS marketplace. Each phase has explicit objectives, activities, artifacts and phase exit criteria. Each phase ends with an anchor point that provides an opportunity to review progress, ensure continued stakeholder commitment to the evolving solution, and to decide to proceed, change direction, or terminate the project. The four phases are, as in RUP, Inception, Elaboration, Construction, and Transition.

- The goal of the Inception Phase is to achieve concurrence among affected stakeholders on the life-cycle objectives for the project. The Inception Phase establishes feasibility through the business case that shows that one or more candidate solutions exist.
- The goal of the Elaboration Phase is to achieve sufficient stability of the architecture and requirements; to select and acquire components; and to mitigate risks so that a single, high-fidelity solution can be identified with predictable cost and schedule.
- The goal of the Construction Phase is to achieve a production-quality release ready for its user community. The selected solution is prepared for fielding.
- The goal of the Transition Phase is to transition the solution to its users. The selected solution is fielded to the user community and supported.

The four EPIC phases are repeated for each solution. Thus across the life of a large or complex project, many solutions–often overlapping–are created and retired in response to new technology, new components, and new operational needs.

Phase Activities and Tasks

While every iteration has the same activities, the emphasis within each activity, and the tasks that support that activity, may change depending on the phase. Figure 5 summarizes the tasks in each of the major iteration activities across the four EPIC phases.

Inception	Elaboration	Construction	Transition
Plan the Iteration			
Build a detailed plan for the iteration			
Update the Development Plan for the project			
Gather...			
... an Understanding of Stakeholder Needs and End-User Business Processes			
Update or create a business model	Update and expand the business model	Update and expand the business model as necessary	
Capture the critical behaviors of the solution	Capture the significant behaviors of the solution	Capture the behaviors of the solution	Update the behaviors of the solution as needed
... an Understanding of Architecture and Design			
Determine architectural context	Amplify the architectural context	Review, and update as needed, the architectural context	Monitor the architectural context
Identify architectural alternatives	Amplify the architectural alternatives contained in solution(s)		
... an Understanding of Marketplace and Other Sources			
Identify relevant component sources	Monitor relevant component sources	Monitor relevant market segments	
Characterize available components	Evaluate applicable components	Characterize component changes	
... an Understanding of the Programmatics and Risks			
Identify management information	Update management information		
Identify procurement needs and opportunities	Update procurement needs and opportunities		
Identify implications of changes to the end user's business process	Amplify implications of changes to end user's business process	Monitor implications of changes to the end user's business process	
Identify risks	Update risks		

Figure 5. Iteration Tasks by Phase

Inception	Elaboration	Construction	Transition
Refine the Understanding of the Solution			
Synthesize information in candidate solutions	Identify and resolve mismatches from the synthesis of new information		
Analyze and negotiate mismatches for each candidate solution			
Characterize each candidate solution	Amplify the solution(s)	Update the solution if needed	
Assemble an Executable Representation			
Build and test proof of concept(s)	Build and test an architectural prototype	Build and test the solution	Build and test releases of the solution
Prototype the needed changes to the end user's business process		Implement the needed changes to the end user's business process	
		Make any needed existing infrastructure and external interfaces changes	
Assess the Iteration			
Assess the Executable Representation	Assess the architectural prototype for the solution(s)	Assess the solution	
Update the information about the solution			
Determine lessons learned from iteration			
Assess the phase, if the iteration completes the phase			Review all phases if the iteration retires the solution.

Figure 5 (continued). Iteration Tasks by Phase

A summary of **Supporting Activities** follows the Iteration Activities in each phase description. Supporting Activities include tasks that must be accomplished during iterations within the phase, but may not be part of every iteration. Some of these activities are summarized in Figure 6. The activities associated with project monitoring and control, and supporting process activities that are described in the Development Plan (such as requirements management or configuration management) are not explicitly addressed here, but are vital to the success of the project.

Inception	Elaboration	Construction	Transition
Monitor project status			
Prepare experimentation facility		Maintain the experimentation facility	
Update and create contracting vehicles as necessary			

Figure 6. Supporting Tasks by Phase

Major Artifacts

The EPIC artifacts are intended to capture the evolving state of the solution and can be thought of as interim work products of the process. Most of the artifacts are started in outline form in the Inception Phase and are expanded across later phases as more information is gathered and refined. The major artifacts used in EPIC and their primary uses are summarized in Figure 7. Artifact names are indicated in the chapters on the individual phases using a special font.

TO CHARACTERIZE END-USER BUSINESS PROCESSES AND STAKEHOLDER NEEDS

- Current Business Use-case Model
- Current Business Object Model
- Target Business Use-case Model
- Target Business Object Model
- Glossary
- Stakeholder Requests
- Solution Requirements Specification
- Use-case Model
- Use Cases
- Supplementary Specification

TO CHARACTERIZE THE MARKETPLACE AND OTHER SOURCES

- Market Segment Information
- Component Dossier (for each examined component)
- Component Screening Criteria and Rationale

TO CHARACTERIZE THE ARCHITECTURE AND DESIGN

- Solution Vision
- Architecture Document
- Design Model
- Executable Representation (s)
  - Implementation Model
- Test Set Artifacts (includes the Test Plan)
- Deployment Artifacts
  - End-user Support Materials (optional in first two phases)
  - Release Notes (required in Transition)
  - Training Materials (required in Transition)
  - Installation Artifacts (required in Transition)

DEVELOPMENT PLAN ELEMENTS TO CHARACTERIZE PROGRAMMATICS AND RISK

Management Process
- Project Plan
- Iteration Plans

Project Monitoring and Control
- Requirements Management Plan
- Schedule Control Plan
- Budget Control Plan
- Quality Control Plan
- Reporting Plan
- Measurement Plan

Risk Management
- Risk Management Plan

Technical Process Plans
- Development Case
- Infrastructure Plan
- Solution Acceptance Plan

Supporting Process Plans
- Configuration Management Plan
- Evaluation Plan
- Documentation Plan
- Quality Assurance Plan
- Problem Resolution Plan
- Vendor/Supplier Relationship Plan
- Process Improvement Plan

ADDITIONAL ARTIFACTS THAT CHARACTERIZE PROGRAMMATICS AND RISK

- Business Case (includes business context, success criteria, financial forecast)
- Business Process Change Management Plan
- Risk List
- Deployment Plan
- License Agreements
- Iteration Assessments (one/iteration)
- Review Record (one/phase)

Figure 7. Master Artifact List

Differences from the RUP

For people who are experienced with the RUP, this section lists some of the major ways that the RUP has been modified for EPIC. This list is not intended to be exhaustive, but to provide highlights of the changes made.

To meet the demands of building, fielding and supporting COTS-based systems, EPIC refined the RUP workflows as follows:

- Requirements must stay fluid until the component implications are understood–often until well within the Elaboration Phase.
- The analysis and design activities must start sooner–in parallel with the requirements activities.
- The project management activities include the management of vendor/supplier relationships.
- Business modeling is mandatory rather than optional.
- Activities are added to monitor the marketplace and evaluate candidate components.
- Business, contracting, and organizational change activities are integrated throughout.

In addition, EPIC expands the RUP in the following areas:

- The Lifecycle Objectives (LCO) anchor point has been redefined to allow multiple candidate solutions to proceed to the Elaboration Phase.
- Iterations are more chaotic to allow for simultaneous gathering and refining of information in all four spheres.
- An experimentation facility is essential across all phases to evaluate new and changed components in the context of the evolving solution.

Readers who refer to RUP sources should note that the RUP use of *system* or *product* is generally equivalent to "solution" in EPIC.

Inception Phase

The Inception Phase establishes a common understanding among stakeholders of the solution consistent with the organization's longer-term objectives. This common understanding, or scope, defines what the solution will do and why. It provides a basis for planning the technical content of each of the following phases and for estimating the cost and time to develop the solution.

The focus of the Inception Phase is on gathering information from each of the four spheres and capturing that information in the form of project artifacts. Most of the artifacts are started in outline form and will be expanded across later phases as more information is gathered and refined. The disparate stakeholders identify the most critical needs and constraints in each sphere. This information includes

- a high-level understanding of the end-user needs, expectations, and constraints. As solutions are defined, stakeholder requests are challenged to ensure that each need and the implication of not meeting that need are fully understood.
- a market survey to understand the makeup, motivations, and components available in the relevant market segment(s). As components are identified, the suppliers of those components are examined for long-term viability, and limited experiments with the components are conducted to evaluate component suitability.
- the constraints imposed by previous solutions, available technology and components as well as applicable standards, external interfaces, and any existing systems with which the solution must interact
- the cost and schedule targets for the project, available procurement vehicles for needed components and services, impediments to end-user business process change, and risk

The gathered information is refined through analysis and negotiation with appropriate stakeholders to form one or more feasible, albeit high-level, candidate solutions. A candidate solution is feasible if it describes a useful capability based on available components that can be integrated into the broader organization's architecture, in a reasonable period of time, at affordable cost, and for acceptable risk. The candidate solutions are summarized in an initial business case where identification of the solutions recommended for detailed examination in the Elaboration Phase is also made. An Executable Representation in each iteration

demonstrates the feasibility of the candidate solutions to meet the critical use cases (the primary scenarios of operation).

Because each candidate solution could contain very different combinations of components, and the components address different aspects of the critical use cases, each candidate solution may represent differing stakeholder needs, architecture, user processes, and programmatic decisions. A manageable number (two to four) of feasible candidates will be carried forward into the Elaboration Phase for further definition of the four spheres.

The Inception Phase ends with the Life-cycle Objectives (LCO) anchor point. Whereas in the RUP, LCO means that the requirements are settled sufficiently to form an architecture; in EPIC, LCO means one or more candidate solutions are identified that meet the high level objectives for the solution and have key stakeholder concurrence. The LCO anchor point reviews the phase exit criteria, determines that the phase objectives have been met, validates stakeholder concurrence on the scope of the solution, and seeks approval to examine the most viable candidate solutions in greater depth.

Phase Objectives

Establish a common understanding of the scope of the solution and its boundary conditions, including interfaces to systems outside the boundary of the solution.

Outline viable candidate architectures.

Differentiate the critical Use Cases of the solution, the primary scenarios of operation that will drive the major tradeoffs.

Identify and evaluate the relevant segments of the commercial marketplace and other sources for components and vendors/suppliers and negotiate tradeoffs among critical Use Cases, the candidate architectures, cost, schedule, and risk.

Exhibit and demonstrate at least one candidate solution against the critical Use Cases.

Estimate cost, schedule and potential risks for each candidate solution.

Determine potential changes to the end user's business process and the tolerance for and inhibitors to implementing those changes across the organization.

Establish a plan for acquiring any components and services needed for this solution.

Phase Task Overview

Phase Planning Activities
- *Plan the Project*

Iteration Activities
- *Plan the Iteration*
  - Build a detailed plan for the iteration
  - Update the Development Plan for the project
- *Gather ...*
- *... an Understanding of Stakeholder Needs and End-User Business Processes*
  - Update or create a business model
  - Capture the critical behaviors of the solution
- *... an Understanding of Architecture and Design*
  - Determine architectural context
  - Identify architectural alternatives
- *... an Understanding of Marketplace and Other Sources*
  - Identify relevant component sources
  - Characterize available components
- *... an Understanding of the Programmatics and Risks*
  - Identify management information
  - Identify procurement needs and opportunities
  - Identify implications of changes to the end user's business process
  - Identify risks
- *Refine the Understanding of the Solution*
  - Synthesize information in candidate solutions
  - Analyze and negotiate mismatches for each candidate solution
  - Characterize each candidate solution
- *Assemble an Executable Representation*
  - Build and test proof of concept(s)
  - Prototype the needed changes to the end user's business process
- *Assess the Iteration*
  - Assess the Executable Representation
  - Update the information about the solution
  - Determine lessons learned from iteration
  - Assess the phase, if the iteration completes the phase

Supporting Activities
- Monitor project status
- Prepare experimentation facility
- Update and create contracting vehicles as necessary

Phase Exit Criteria

Status	Exit Criteria
	Affected stakeholders concur that the scope of each of the candidate solutions is feasible and represents a useful capability.
	Critical mismatches between stakeholder needs and component capabilities are negotiated and represented in critical use cases.
	Cost/schedule estimates, project tasks, risk, and engineering processes for each candidate solution are credible.
	The risks for each candidate solution are understood, fall within an acceptable range, and mitigations are identified for critical risks.
	The potential changes to the end user's business process have been identified for each candidate solution, with stakeholder agreement to implement the changes, should the solution be selected.
	The depth and breadth of an Executable Representation (e.g., mockup, architecture prototype) demonstrates the defined scope of each candidate solution.
	The project has initiated relationships with key vendors and suppliers to provide needed insights into component capabilities and directions.
	Any differences between actual resource expenditures versus planned expenditures for this phase are understood, and corrective actions are included in the Project Plan.
	The Development Plan has been updated. (The plan for the Elaboration Phase is sufficiently detailed and accurate, and is backed up with a credible basis for all estimates.)
	The experimentation facility is sufficient to evaluate the impact of candidate components and the candidate solutions within the broader context of the organization's architecture in the Elaboration Phase.
	The License Agreements, contracts, and procurement vehicles for needed components and services are in place for the Elaboration Phase.

Elaboration Phase

LCO marks a change in intensity. The basic activities for the Elaboration Phase are the same as those in the Inception Phase, but the level of detail is deeper and the level of resource commitment is significantly higher. The Elaboration Phase achieves sufficient understanding and stability of the requirements, end-user business process, and architecture; selects and acquires components; and mitigates risks such that a single solution is identified that meets a valid organizational need with acceptable cost and schedule.

The focus of the Elaboration Phase is on in-depth hands-on experiments with the candidate solutions by end users and engineers. These experiments and prototypes are conducted in an experimentation facility that represents, as closely as practical, the operational environment. This phase includes further definition of stakeholder needs and end-user business processes based on detailed evaluation of the components. This phase also includes definition and prototyping of the strategy and mechanisms for component integration, data migration and component tailoring.[15] The focus continues to be keeping the four spheres in balance as greater knowledge of each of the candidate solutions is gained.

When the candidate solutions are sufficiently understood, one solution is selected that will become the basis for the Construction Phase. The selected solution is further amplified, using the experimentation facility, until it is shown that the selected solution has achieved sufficient stability in requirements and architecture as demonstrated in an Executable Representation.

The Elaboration Phase ends with the Life-cycle Architecture (LCA) anchor point when all stakeholders agree that the solution provides sufficient operational value to stakeholders and can be assembled by the engineers for acceptable cost, schedule, and risk. At this point, all components have been selected and procured, any integration mechanisms to incorporate the components and any other components are validated, and the risk, cost, and schedule for completion of the project have been predicted within an acceptable range.

[15] *Tailored* means non-source code adjustment necessary to integrate the COTS products into an operational system (e.g., scripts).

Phase Objectives

Baseline one high-fidelity solution for implementation in the Construction Phase.

- Baseline the Solution Vision
- Define, validate, and baseline the architecture including component integration mechanisms
- Capture the functional and non-functional requirements
- Understand the relevant components
- Agree upon needed changes to the end user's business process

Demonstrate that the baseline solution (including components and their integration mechanisms) will support the Solution Vision at a reasonable cost in a reasonable time.

Develop a detailed Business Process Change Management Plan to implement the agreed-upon changes to the end user's business process.

Acquire the components and services needed for the Construction Phase and initial fielding in the Transition Phase.

Phase Task Overview

Phase Planning Activities
- Update the Development Plan for the project

Iteration Activities

Plan the Iteration
- Build a detailed plan for the iteration
- Update the Development Plan for the project

Gather ...

... an Understanding of Stakeholder Needs and End-User Business Processes
- Update and expand the business model
- Capture the significant behaviors of the solution

... an Understanding of Architecture and Design
- Amplify the architectural context
- Amplify the architectural alternatives contained in solution(s)

... an Understanding of Marketplace and Other Sources
- Monitor relevant component sources
- Evaluate applicable components

... an Understanding of the Programmatics and Risks
- Update management information
- Update procurement needs and opportunities
- Amplify implications of changes to the end user's business process
- Update risks

Refine the Understanding of the Solution
- Identify and resolve mismatches from the synthesis of new information
- Amplify the solution(s)

Assemble an Executable Representation
- Build and test an architectural prototype
- Prototype the needed changes to the end user's business process

Assess the Iteration
- Assess the architectural prototype for the solution(s)
- Update the information about the solution
- Determine lessons learned from iteration
- Assess the phase if the iteration completes the phase

Supporting Activities
- Monitor project status
- Prepare experimentation facility
- Update and create contracting vehicles as necessary

Phase Exit Criteria

Status	Exit Criteria
	Affected stakeholders agree that within acceptable risk, the vision of one solution can be met, in the context of the solution architecture, if the current plan to build the solution is executed. ▪ The requirements are identified and negotiated to optimize leverage of the marketplace and other sources. ▪ The architecture is stable and reflects a structure that is flexible enough to support the continuing evolution of components and operational needs. ▪ The stakeholders, who are subject to the changes, support the necessary changes to the end user's business process. ▪ Cost/schedule estimates and project tasks are credible.
	An Executable Representation demonstrates the common understanding of the solution that has been achieved through negotiation with stakeholders, and addresses (and resolves as appropriate) major risks of the solution.
	The remaining risks are understood, are acceptable and have appropriate mitigations identified.
	The Business Process Change Management Plan realistically accounts for all types of end users and the necessary changes for both individuals and their organizations.
	The project has defined and implemented relationships with key vendors/suppliers that provides needed insights into component capabilities and directions. (In the Vendor/Supplier Relationship Plan)
	Any differences between actual resource expenditures versus planned expenditures for this phase are understood and corrective actions are included in the Project Plan.
	The experimentation facility and the Development Case are sufficient to support beginning the Construction phase.
	The Development Plan has been updated. (The plan for the Construction Phase is sufficiently detailed and accurate and is backed up with a credible basis for all estimates.)
	The License Agreements, contracts, and procurement vehicles for needed components and services are in place for the Construction Phase and initial fielding.

Construction Phase

The focus of the Construction Phase is on preparation of a production-quality release of the selected solution approved at the LCA anchor point that is suitable for fielding. Any custom components needed are developed. Production rigor is applied to component tailoring, integration code or data (including wrappers, glue, data sets, etc.) needed to incorporate pre-existing and custom components, and system testing. Additionally, the Construction Phase includes preparation of necessary support materials, such as installation instructions, version descriptions, user and operator manuals, and other user and installation site support required.

The Construction Phase continues the preparation of the end-user business environment of the target organizations to facilitate the initial fielding of the solution. This preparation includes development of required policies and procedures, restructuring of the organization as necessary, implementation of changes to the end user's business process for the initial rollout groups, and the establishment of incentives, user groups, and other mechanisms to encourage adoption of the solution.

While every effort is made during the Elaboration Phase to stabilize the solution and to address risks, inevitably some unanticipated changes may occur in requirements, components, and the architecture and design during the Construction Phase. In particular, because of the volatile nature of the marketplace, new versions of the selected components will require detailed investigation as suppliers add, change or remove functionality. Continued monitoring of the marketplace and evaluation of new and changed components is required to anticipate changes and determine an appropriate component upgrade approach.

The Construction Phase ends with the Initial Operational Capability (IOC) anchor point. The IOC anchor point allows stakeholders to verify that a production-quality release of the solution is ready for fielding to at least a subset of the operational users as an initial fielding or beta test.

Phase Objectives

Implement the selected solution with production quality to operate within the broader organization's architecture.

Minimize engineering costs by optimizing resources and avoiding unnecessary scrap and rework.

Achieve adequate quality as rapidly as practical.

Achieve useful versions (alpha, beta, and other test releases) as rapidly as practical.

Maintain current marketplace information.

Balance engineering stability and potential component obsolescence with marketplace volatility.

Prepare initial fielding end users for beta test (e.g., implement, or prepare to implement, changes to the end-users' business process; complete training; manage resistance).

Put contract vehicles in place for the Transition Phase.

Phase Task Overview

Phase Planning Activities
- Update the Development Plan for the project

Iteration Activities

Plan the Iteration
- Build a detailed plan for the iteration
- Update the Development Plan for the project

Gather ...

... an Understanding of Stakeholder Needs and End-User Business Processes
- Update and expand the business model as necessary
- Capture the significant behaviors of the solution

... an Understanding of Architecture and Design
- Review, and update as needed, the architectural context

... an Understanding of Marketplace and Other Sources
- Monitor relevant market segments
- Characterize component changes

... an Understanding of the Programmatics and Risks
- Update management information
- Update procurement needs and opportunities
- Monitor implications of changes to the end user's business process
- Update risks

Refine the Understanding of the Solution
- Identify and resolve mismatches from the synthesis of new information
- Update the solution if needed

Assemble an Executable Representation
- Build and test the solution
- Implement the needed changes to the end user's business process
- Make any needed existing infrastructure and external interfaces changes

Assess the Iteration
- Assess the Solution
- Update the information about the solution
- Determine lessons learned from iteration
- Assess the phase, if the iteration completes the phase

Supporting Activities
- Monitor project status
- Maintain the experimentation facility
- Update and create procurement vehicles as necessary

Phase Exit Criteria

Status	Exit Criteria
	Affected stakeholders agree the solution baseline, as demonstrated in the Executable Representation, is mature enough to be fielded in the user community (the release is stable). ▪ Existing defects are not obstacles to achieving the purpose of the release. ▪ Pending changes are not obstacles to achieving the purpose of the release.
	Affected stakeholders approve the defined and documented procedures for transition of the solution to the user community (and have implemented the procedures for users affected by initial fielding).
	Relationships with vendors are adequately managed.
	Information on relevant components and the marketplace is current and recorded.
	Any differences between actual resource expenditures versus planned expenditures for this phase are understood, and corrective actions are included in the Project Plan.
	The experimentation facility is sufficient to support continued monitoring of the components and relevant market segments.
	The Development Plan has been updated. (The plan for the Transition Phase is sufficiently detailed and accurate and is backed up with a credible basis for all estimates.)
	The Deployment Plan is sufficient to support beginning the Transition phase.
	The contract vehicles are in place for initial fielding and in progress for full fielding.

Transition Phase

The Transition Phase is focused on moving the solution to the user community. This requires that the users attain proficiency in the solution and the end-user business processes that the solution supports, are motivated to use the solution, and are self-supporting in their use of the solution.

The Transition Phase begins with an initial fielding, or beta test of the solution developed in the Construction Phase. Following a decision to make the solution release generally available, the solution will be fielded across the user base. As required, bugs are fixed, features are adjusted, and missing elements are added to the fielded solution in maintenance releases. Continued monitoring of the marketplace and other sources is required to anticipate changes. Maintaining an experimentation facility for component evaluation to assess the potential impact of any new or changed components is essential.

The Transition Phase encompasses continued support for the solution. The Transition Phase ends when the solution is retired and replaced by a new solution. The activities of this phase are required for support of the solution even if support is provided by an organization different from the organization responsible for its implementation. In this case, it is incumbent on the implementation organization to transfer the knowledge that has been gained in the previous phases and iterations to the support organization.

Phase Objectives

Achieve stakeholder concurrence that the baselines for fielding are complete and consistent with the acceptance criteria based on the Solution Vision.

Implement changes to the end user's business process changes across the user community.

Achieve user satisfaction.

Achieve user self-supportability (e.g., procedures in place, training complete, maintenance plan in place).

Achieve final solution baselines as rapidly and cost effectively as is practical.

Maintain current information about the marketplace and other component sources.

Maintain adequate visibility into component changes through managing relationships with vendors/suppliers.

Balance solution stability with marketplace volatility.

Position procurement vehicles for full fielding and long-term support of the solution.

Collect, analyze, and make accessible for future use information relating to the conduct of the EPIC process.

Phase Task Overview

Phase Planning Activities
- Update the Development Plan for the project

Iteration Activities

Plan the Iteration
- Build a detailed plan for the iteration
- Update the Development Plan for the project

Gather ...

... an Understanding of Stakeholder Needs and End-User Business Processes
- Update and expand the business model as necessary
- Update the behaviors of the solution as needed

... an Understanding of Architecture and Design
- Monitor the architectural context

... an Understanding of Marketplace and Other Sources
- Monitor relevant market segments
- Characterize component changes

... an Understanding of the Programmatics and Risks
- Update management information
- Update procurement needs and opportunities
- Monitor implications of changes to the end user's business process
- Update risks

Refine the Understanding of the Solution
- Identify and resolve mismatches from the synthesis of new information
- Update the solution if needed

Assemble an Executable Representation
- Build and test releases of the solution
- Implement the needed end-user business processes
- Make any needed existing infrastructure and external interfaces changes

Assess the Iteration
- Assess the Solution
- Update the information about the solution
- Determine lessons learned from iteration
- Review all phases if the iteration retires the solution

Supporting Activities
- Monitor project status
- Maintain experimentation facility
- Update and create contracting vehicles as necessary

Phase Exit Criteria

Status	Exit Criteria
	Solution functionality is no longer needed or is replaced by a succeeding solution.
	Improvements to, lessons learned, and other useful information on the defined implementation process, tools, and methods are collected and made accessible to future projects.
	Information from the project is reviewed, analyzed, and used to improve the organization's standard implementation process.

Glossary

activity
A unit of work that a worker may be asked to perform. [12]

architecture
A high-level design that provides decisions made about the problem(s) that the component will solve, component descriptions, relationships between components, and dynamic operation description. [13]

A description of the essential elements of the system and the relationships between them. The elements include structure, behavior, usage, functionality, performance, resilience, reuse, comprehensibility, economic and technologic constraints and tradeoffs, and aesthetic issues. [12]

artifact
A piece of information that is produced, modified, or used by a process, defines an area of responsibility, and is subject to version control. An artifact can be a model, a model element, or a document. [12]

baseline
A reviewed and approved release of artifacts that constitutes an agreed-on basis for evolution or construction and that can be changed only through a formal procedure, such as change and configuration control. [12]

business process
The tasks, duties, or functions performed to support the objectives of an organization. [12]

commercial item
An item customarily used for nongovernmental purposes that has been or will be sold, leased, or licensed (or offered for sale, lease, or license) to the general public. An item that includes modifications customarily available in the commercial marketplace or minor modifications made to meet federal government requirements is still a commercial item. Services such as installation, maintenance, repair, and training, procured for support of an item described above are considered commercial items if they are offered to the public

under similar terms and conditions or sold competitively in substantial quantities based on established catalog or market prices. [14]

commercial off-the-shelf (COTS) component
A component that is (a) sold, leased, or licensed to the general public, (b) offered by a vendor trying to profit from it, (c) supported and evolved by the vendor, who retains the intellectual property rights, (d) available in multiple, identical copies; used without modification of the internals. [6]

component
A nontrivial, nearly independent, and replaceable part of a system that fulfills a clear function in the context of a well-defined architecture. A component conforms to and provides the physical realization of a set of interfaces. [12]

In EPIC, the term "component" is used for one or more pre-existing hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (piece of the system being replaced), reuse libraries, or other reuse sources (e.g., free-ware, shareware).

constraint
(1) A restriction, limit, or regulation. (2) A type of requirement that cannot be traded against other requirements. [13]

COTS-based system
Systems that depend on the use of one or more commercial off-the-shelf (COTS) parts to meet specific business needs. These systems can also include items developed for other entities, legacy components, and custom code. COTS-based system denotes a spectrum of systems: those that can be formed from one substantial COTS component that is tailored to provide significant system functionality (a COTS-Solution System) or those formed from integrating multiple components supplied by a variety of sources to collectively provide functionality (a COTS-Aggregate System).

custom
Made to order. Uniquely built in direct response to a buyer's specifications.

customer
A purchaser or user of end components. [13]

One who buys goods or services. [15]

design
The part of the engineering process whose primary purpose is to decide *how* the system will be implemented. During design, strategic and tactical decisions are made to meet the required functional and quality requirements of a system. [12]

The set of decisions about a component that results in a common vision of what need it addresses, and how it addresses or satisfies that need. Typically, a design includes an operational concept (how users are expected or intended to use the component), components and their relationships, and sometimes decisions about the processes that will produce, field, and support it. [13]

end users
Those who will be using the system in the operational environment.

An individual or organization that uses, applies, or operates an end or enabling component. [13]

engineering
The integrated technical effort responsible for solution development that balances all factors associated with meeting system life-cycle requirements. [13]

evolutionary prototype
A prototype that evolves from one iteration to the next to become the final solution. Evolutionary prototypes tend to be designed rather formally and tested somewhat formally, even in early stages. Evolutionary prototypes are likely to use the infrastructure of the ultimate solution. [12]

exit criteria
Specific accomplishments or conditions that must be satisfactorily demonstrated before an effort can progress further in the current life-cycle phase or transition to the next phase. [13]

experimentation facility
Represents, as closely as practical, the operational environment in which hands-on experiments with components and/or a solution are conducted by engineers and end users.

feasible
A candidate solution is feasible if it describes a useful capability that can be integrated into the broader organization's architecture, in a reasonable period of time, at affordable cost, for acceptable risk, and based on available components.

fielding
Moving a version of the solution into some part of the end-user community.

harmonized
A solution is *harmonized* at a given level of abstraction when the candidate components can be assembled according to the evolving design to fulfill the agreed-upon requirements to support end-user business processes with acceptable cost, schedule, and risk.

Initial Operational Capability (IOC)
The Construction Phase ends with the Initial Operational Capability (IOC) anchor point. The IOC anchor point allows stakeholders to verify that a production-quality release of the

solution is ready for fielding to at least a subset of the operational users as an initial fielding or beta test. [16]

integration
The engineering activity in which components are combined into an executable whole. [12]

The merger or combining of two or more elements (e.g., components, parts, or configuration items) into a functioning and higher level element with the functional and physical interfaces satisfied. [13]

iteration
A distinct sequence of activities with a baselined plan and valuation criteria resulting in a release (internal or external). [12]

knowledge
Includes an increasingly detailed understanding of (a) the capabilities and limitations of candidate components, (b) the implications of the components on the requirements for the solution and the end user's business processes, as well as the planning necessary to implement any needed changes, and (c) the architectural alternatives and integration mechanisms that bind the components together.

life cycle
The scope of systems beginning with the identification of a perceived customer need, addressing engineering, test, manufacturing, operation, support and training activities, and continuing through various upgrades or evaluation until the component disposal. [13]

Life-cycle Architecture (LCA)
The Elaboration Phase ends with the Life-cycle Architecture (LCA) anchor point milestone. Management verifies the basis for a sound commitment to development and evolution of a particular architecture that is shown to be feasible with respect to budget, schedule, requirements, operations concept, and business case and the elimination of all critical risk items. [16]

For EPIC at this point, all components have been selected and procured and any integration mechanisms necessary to incorporate the pre-existing components and any other components are validated.

Life-cycle Objective (LCO)
The Inception Phase ends with the Life-cycle Objectives (LCO) anchor point milestone. Management verifies the basis for a business commitment to proceed. [16]

In the RUP, LCO means that the requirements are settled sufficiently to form an architecture; in EPIC, LCO means one or more candidate solutions are identified that meet the project's high-level objectives and have key stakeholder concurrence. The LCO anchor point reviews the phase exit criteria, determines that the phase objectives have been met,

validates stakeholder concurrence on the scope of the solution, and seeks approval to examine the most viable candidate solutions in greater depth.

marketplace
The aggregation of buyers and sellers where goods are offered for sale, lease, or license.

model
A semantically closed abstraction of a system. In the RUP, a complete description of a system from a particular perspective–"complete"–meaning that you don't need additional information to understand the systems from that perspective. [12]

A simplified representation of some aspect of the real world. [13]

need
A user-related capability shortfall (such as those documented in a need statement, field deficiency report, or engineering change order), or an opportunity to satisfy a new market or capability requirement because of a new technology application or breakthrough, or to reduce costs. Needs may also relate to providing a desired service (e.g., system disposal). [13]

non-functional
Attributes of the solution that impose conditions on functional needs or requirements. Non-functionals address issues such as

- usability (e.g., human factors, aesthetics, consistency in the user interface, on-line help, user documentation, training materials)
- reliability (e.g., frequency/severity of failure, recoverability, predictability, accuracy, mean time between failure)
- performance (e.g., speed, efficiency, availability, accuracy, throughput, response time, recovery time, resource usage)
- supportability (e.g., testability, extensibility, adaptability, maintainability, compatibility, configurability, serviceability, installability, internationalization)
- physical characteristics (e.g., material used, shape, size, weight) [13]

phase
The time between two major project milestones during which a well-defined set of objectives is met, artifacts are completed, and decisions are made to move or not to move into the next phase. [12]

plan
A documented series of tasks required to meet an objective, typically including the associated schedule, budget, resources, organizational description and work breakdown structure. [13]

pre-existing component
One or more pre-existing hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware).

procurement
The act of buying, leasing, or licensing components and/or services.

production quality
A version of the solution that is built with sufficient quality for use in the end-user's operational environment.

project
An effort consisting of both technical and management activities for the purpose of building, fielding, and supporting a solution.

prototype
An executable. Behavioral prototypes focus on exploring specific behaviors of the system. Structural prototypes explore architectural or technological concerns. Exploratory prototypes are thrown away after they are finished and you have learned whatever you wanted from them. Exploratory prototypes are not necessarily subject to change management and configuration control. Evolutionary prototypes evolve to become the final system. As such, they may be subject to change management and configuration control. [12]

A model (physical, electronic, digital, analytical, etc.) of a solution built or constructed for the purpose of: (a) assessing the feasibility of a new or unfamiliar technology (b) assessing or mitigating technical risk (c) validating requirements (d) demonstrating critical features (e) qualifying a system (f) qualifying a process (g) characterizing performance or component features, or (h) elucidating physical principles. [13]

release
A subset of the end component that is the object of evaluation at a major milestone. [12]

requirement
A description of a condition or capability of a system that must be present; either derived directly from user needs or stated in a contract, standard, specification, or other formally imposed document. [12]

Something that governs that a component will have a given characteristic or achieve a given purpose, including what, how well and under what conditions. [13]

reuse component
One or more pre-existing hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware).

risk
An ongoing or impending concern that has a significant probability of adversely affecting the success of major milestones. [12]

solution
A solution provides useful capability that can be fielded in a period of six to twelve months. A solution is the integrated assembly of

- one or more pre-existing hardware and software components from the commercial marketplace (i.e., COTS components), the legacy system (piece of the system being replaced), reuse libraries, or other reuse sources (e.g., freeware, shareware)
- any required custom code (including wrappers and "glue")
- appropriate linkage/interface to the broader organization's architecture and external systems
- end user's business process including any changes necessary to match the processes provided by the integrated components and custom code

spheres of influence
The term used to represent a set of information with common or related stakeholders, techniques for gathering and managing the information, and dynamic by which the information changes. EPIC focuses on four spheres of influence in forming solutions: 1) stakeholder needs and end-user business processes, 2) architecture and design, 3) the commercial marketplace and other source, and 4) management of the project, planning and implementation for any needed changes to the end user's business process, and continuous refinement of the project's cost, schedule, and risk. An emphasis on balance between the four spheres is critical to EPIC and must continue throughout the life of a project.

stakeholder
Any person or representative of an organization who has a stake–a vested interest–in the outcome of a project or whose opinion must be accommodated. A stakeholder can be an end user, a purchaser, a contractor, a developer, or a project manager. [12]

An individual or organization interested in the success of a component or system. Examples of stakeholders include customers, developers, engineers, managers, manufacturers, and end users, etc. [13]

suppliers
Providers of components that are not COTS components.

tailoring
Using mechanisms that a supplier builds into a component to allow it to meet the specific needs of a particular system. Tailoring does not involve changes to the internal aspects of the component, such as source code. [7]

use case
A sequence of actions a system performs that yields an observable result of value to a particular actor. A use case contains all main, alternative, and exception flows of events related to producing the observable result of value. [12]

vendor
A commercial enterprise that produces and maintains COTS components. [7]

vision
The user's or customers' view of the component to be developed. [12]

References

1. Boehm, B. & Abts, C. "COTS Integration: Plug and Pray?" *IEEE Computer, 32*, 1 (January 1999) 135-140.

2. Garlan, D.; Allen, R.; & Ockerbloom, J. "Architecture Mismatch: or Why It's Hard to Build System out of Existing Parts." *Proceedings of the International Conference on Software Engineering.* Seattle, WA, April 23-20, 1995.

3. Brownsword, L.; Carney, D.; & Oberndorf, P. "The Opportunities and Complexities of Applying COTS Components." *Crosstalk, 11*, 4 (April 1998). http://www.stsc.hill.af.mil/crosstalk/1998/apr/applying.asp

4. Office of the Secretary of Defense, *Commercial Item Acquisition: Considerations and Lessons Learned.* http://www.acq.osd.mil/ar/doc/cotsreport.PDF (26 June 2000).

5. United States Air Force Science Advisory Board report on Ensuring Successful Implementation of Commercial Items in Air Force Systems, SAB-TR-99-03. Available through the 1999 archived reports link at http://www.sab.hq.af.mil/archives/reports/ (April 2000).

6. Oberndorf, T.; Brownsword, L.; & Sledge, C. *An Activity Framework for COTS-Based Systems*, (CMU/SEI-2000-TR-010 ADA383836). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2000. http://www.sei.cmu.edu/publications/documents/00.reports/00tr010.html

7. Meyers, B.C. & Oberndorf, P. *Managing Software Acquisition: Open Systems and COTS Products.* New York, NY: Addison-Wesley, SEI Series in Software Engineering, 2001.

8. Carney, D.; Hissam, S.; & Plakosh, D. "Complex COTS-based Software Systems: Practical Steps for their Maintenance." *Journal of Software Maintenance: Research and Practice, 12*, 6 (2000): 357-376.

9. Wallnau, K.C.; Hissam, S.A.; and Seacord, R.C. *Building Systems from Commercial Components.* New York, NY: Addison-Wesley, SEI Series in Software Engineering, 2002.

10. Kruchten, Phillippe. *The Rational Unified Process: An Introduction,* 2nd ed. New York, NY: Addison-Wesley Object Technology Series, March 2000.

11. Boehm, B. "A Spiral Model of Software Development and Enhancement." *IEEE Computer,* 21, 2 (February 1998): 61-72.

12. Rational Unified Process, (software product) version 2000.02.10. Rational Software Corporation. http://www.rational.com/

13. *Interim Standard for Systems Engineering Capability Model.* EIA/IS 731.1, 1996. http://www.geia.org/sstc/G47/731dwnld.htm

14. *Federal Acquisition Regulations – Part 2.* Washington, DC: General Services Administration, 2001. http://www.arnet.gov/far/current/pdf/FAR.book.pdf

15. Webster, Noah. *Webster's II New College Dictionary.* Boston, MA: Houghton Mifflin Company, 1995.

16. Boehm, B. "Anchoring the Software Process." *IEEE Software,* (July 1996): 73-82.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	July 2002	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview Key Elements in Building, Fielding, and Supporting Commercial-off-the-Shelf (COTS) Based Solutions	F19628-00-C-0003

6. AUTHOR(S)
Cecilia Albert, Lisa Brownsword

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2002-TR-009

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116	ESC-TR-009

11. supplementary notes

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

Government and private organizations are escalating their use of commercial off-the-shelf (COTS) and other pre-existing components in critical business systems. Attempts to exploit these components through use of traditional engineering approaches that involve defining requirements, formulating an architecture, and then searching for components that meet the specified requirements within the defined architecture have been disappointing.

The Evolutionary Process for Integrating COTS-based systems (EPIC) redefines acquisition, management, and engineering practices to more effectively leverage the COTS marketplace and other sources of pre-existing components. This is accomplished through concurrent discovery and negotiation of diverse spheres of influence: user needs and business processes, applicable technology and components, the target architecture, and programmatic constraints. EPIC codifies these practices in a structured flow of key activities and artifacts. This alternative approach is a risk-based, disciplined, spiral-engineering approach which leverages the Rational Unified Process (RUP).

This document is the first release of an overview of the EPIC framework along with its activities and artifacts. The first release of the full description of EPIC is found in the Software Engineering Institute technical report: CMU/SEI-2002-TR-005. These documents will be updated based on reader's comments and lessons learned from use of EPIC.

14. SUBJECT TERMS	15. NUMBER OF PAGES
evolutionary process, integrating COTS-based systems, EPIC, information technology, ITSEP, Rational Unified Process, RUP, inception, elaboration, construction, transition, phase, iteration , spiral development, commercial	58

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18 298-102